Pre-Trained Variational Autoencoder Approaches for Generating 3D Objects from 2D Images

Zafer Serin, Uğur Yüzgeç, and Cihan Karakuzu

1 Introduction

Systems created by humans today are equipped with the ability to think, make decisions, generate ideas, and express emotions [1]. While some studies aim to classify and distinguish objects, others aim to develop freedom of choice and decision-making ability in the face of events and situations. This concept, first introduced in 1956 and known as "artificial intelligence," is gaining popularity and being applied in various fields [2]. Artificial intelligence is a multifaceted concept comprising various sub-fields, including machine learning, object recognition, natural language processing, autonomous systems, advanced data mining, and computational intelligence. Machine learning's objective is to decipher data and discover undiscovered relationships [3]. Although it draws on data analysis and feature extraction processes by humans, machine learning adopts a less human-dependent approach. Many learning algorithms are employed in the field of artificial intelligence. Some of these are decision trees, support vector machines, random forests, and Bayesian models. However, artificial neural networks are widely favored for their capability to learn from extensive datasets and manage complexity [4].

Artificial intelligence, machine learning, and deep learning fields have made significant strides in various sectors and are continuously being developed and adapted for previously untapped or challenging domains [5]. One such domain is imaging methods and computer-aided modeling, which have faced inherent

Z. Serin (✉)
Bilecik Seyh Edebali University, Pazaryeri Vocational School, Bilecik, Turkey
e-mail: zafer.serin@bilecik.edu.tr

U. Yüzgeç · C. Karakuzu
Bilecik Seyh Edebali University, Computer Engineering Department, Bilecik, Turkey
e-mail: ugur.yuzgec@bilecik.edu.tr; cihan.karakuzu@bilecik.edu.tr

M. N. Seyman (ed.), *2nd International Congress of Electrical and Computer Engineering*, EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-031-52760-9_7

challenges since their inception. Ongoing studies strive to overcome these difficulties. One of the most significant challenges is the difficulty of representing 3D objects on a 2D computer screen. Many software solutions have been developed to address this issue and are widely used today. With the gaming industry surpassing both the music and film sectors [6], the demand for 3D objects is also increasing. The demand for 3D objects is rapidly rising, not only in video games but also in fields including movies, virtual reality, augmented reality, architectural design, industrial design, and metaverses. As artificial intelligence advances and the need for 3D objects increases, it is becoming increasingly necessary to incorporate and tailor artificial intelligence and related technologies in the realm of 3D modeling.

The presentation of 3D objects in a computerized setting can be classified into three sub-categories: voxel, point cloud, and polygon [7]. Voxel, among them, can be viewed as an extension of a 3D pixel, i.e., a volumetric pixel. Simply, adding x and y-axis pixels in a 2D environment is equivalent to including a z-axis and thus converting a point with x and y coordinates into a cube. The point cloud approach involves presenting data obtained from sensors in a 3D environment using laser scanning, 3D cameras, and similar tools. Meanwhile, the polygon method represents object surfaces with triangles, which are 2D surfaces with three points in 3D space. These triangles are combined to form more complex surfaces. These techniques are all used to represent 3D objects. Voxel-based object representation has been employed in the present study due to its appropriateness for implementation in artificial neural networks.

With the rise in demand for 3D items and the advancement of artificial intelligence, certain models are being developed for 3D object generation. The ability to generate a 3D object from a single 2D image has various implications. Such an output can serve as an initial representation for those involved in 3D modeling, as well as being applicable in architectural modeling, video gaming, and movie production. In this study, the 3D-VAE-GAN models, which combine the GAN [8] and VAE [9] models, are employed with a voxel-based representation to address the problem. The VAE model's encoder part maps the input data to the latent space and can restore this mapped data to its initial state, preserving information about the original data. In a typical 3D-VAE-GAN model, the encoder can operate as a CNN model. In this study, we evaluate feature extraction and mapping to the latent space using pre-trained encoder networks. We compare different pre-trained networks (DenseNet121 [10], EfficientNetB0 [11], RegNet16 [12], and ResNet18 [13]) to analyze their performances. Additionally, we present the results obtained when using a standard encoder network without the fully connected layer and with the fully connected layer.

1.1 Related Works

In this subsection, some studies in the literature related to the subject of this study are discussed. 3D-GAN, developed in 2016 by Jiajun Wu and his team, is regarded as a

crucial advancement in 3D object manufacturing. This artificial intelligence model employs both GAN and volumetric convolutional neural network structures to create entirely new objects from pre-existing datasets. Additionally, the novel 3D-VAE-GAN model is proposed for the creation of 3D objects using 2D images. Both models utilize voxels as their bases. Based on the evaluations conducted on the ModelNet [14] dataset, it is evident that these models deliver exceptional accuracy rates.

In 2017, Edward J. Smith and David Meger presented a study on the 3D-GAN model. Their work employed gradient penalization and the Wasserstein distance function with the objective of generating successful results across not only single classes but also multiclass generations. The direct object generation work is referred to as 3D-IWGAN, while 3D-VAE-IWGAN pertains to the generation of 3D objects from 2D images. These voxel-based models underwent testing on the IKEA dataset [15] and produced good precision outcomes [16].

In 2018, Jianwen Xie, Zilong Zheng, and colleagues proposed the 3D DescriptorNet, which is a model for generating 3D objects that includes a deep convolutional energy-based neural network. This model is based on voxelized data and uses a probability density function signal to operate. The probabilistic distribution is defined using Markov Chain Monte Carlo, and the 3D object pattern is then derived from this probabilistic distribution. It is a versatile approach to addressing issues such as object recovery and object resolution enhancement. The approach was tested on reconstruction and classification problems using the ModelNet10 dataset, yielding successful outcomes [17].

In 2019, Hui Huang, Zhijie Wu, et al. introduced SAGNet, a model that can learn the relationships between objects and their parts, as well as the connections between parts of different objects, in a structure-sensitive manner. The model incorporates an autoencoder that can learn the geometry and structure of objects and be evaluated using a voxel-based approach with a set of semantically meaningful parts. The reconstruction results obtained by the model were highly successful [18].

In 2020, Yanran Guan, Tansin Jahan, and Oliver van Kaick proposed a 3D-GAE model for generating 3D objects using the generalized autoencoder (GAE) technique. This model is characterized by fast training and high generalization capability. 3D-GAE employs a combination of a standard autoencoder loss function and a unique loss function. It calculates the generated objects' similarity using the Chamfer Distance method and quantifies successful outcomes. The tests on the COSEG [19–21] and ModelNet 40 datasets yielded an effective baseline [22].

In 2020, Rundi Wu, Yixin Zhuang, and colleagues put forward PQ-NET for 3D object generation. This model is founded on the sequential generation of object parts to create an object. This method is akin to constructing a sentence, where, just as the position of words is determined during sentence construction, a specific sequence must be followed and produced sequentially when generating objects. Based on a part-based analysis, PQ-NET employs PartNet, a variant of the ShapeNet dataset with semantically separated objects. PQ-NET, featuring a voxel-based approach, has undergone assessment under the IoU metric for reconstruction and has attained favorable outcomes [23].

2 3D-VAE-GAN Model

In this study, the 3D-VAE-GAN models were utilized to generate 3D objects from 2D images. The voxel object representation method was chosen due to its compatibility with deep learning models, fast processing, boundary recognition, and direct usage with raw data, making it particularly advantageous [24]. The 3D-VAE-GAN architecture combines VAE and GAN models. Standard autoencoder models comprise encoder and decoder neural networks. The encoder network maps the provided input to a latent space of a specific size, while the decoder network aims to replicate the original input by retrieving data from this latent space. While autoencoders are primarily used for operations such as data compression, VAE models are employed for data generation. The main contrast between these models, which are highly alike, is that VAE utilizes a certain probability distribution for encoding data.

On the contrary, GAN is a method comprising two neural networks, a generator and a discriminator neural network, and is based on competitive learning. The generator network in the GAN models is fed input either randomly or adhering to certain norms and attempts to generate an output with specific characteristics and size. The generated output is inputted to the discriminator network, which is another part of GAN and has received prior training with real and fake data. Its task is to distinguish (classify) whether the data is real or fake. As the model's training progresses, the generator network produces more precise data, and the discriminator network becomes adept at distinguishing the generated data more accurately. Essentially, the 3D-VAE-GAN model consists of three neural networks. These are encoder, generator, and discriminator neural network models. Instead of the decoder component in the VAE model, the GAN model incorporates the generator component. The encoder part of the VAE model takes $224 \times 224 \times 3$ images as input, which are then converted into latent space vectors consisting of 200 elements. The images utilized in this study were obtained from the ShapeNet [25] image dataset and featured chairs as objects of interest. Some of these images are illustrated in Fig. 1.

These vectors, which have been mapped to the latent space, are utilized for input into the generator network with the objective of producing the 3D object in the image. The generated objects are subsequently subjected to the classification process through the discriminator network, and the performance of the model is assessed by calculating the average loss values for all three networks. Kullback–Leibler divergence is employed in this study to measure the reconstruction error. Figure 2 exhibits a standard 3D-VAE-GAN model.

The generator used includes five layers. The first four layers comprise transpose convolution, batch normalization, and ReLU layers, whereas the last layer involves a sigmoid layer directly following the transpose convolution process. The kernel size was set to 4, and a stride of 2 was used throughout all layers, with (1,1,1,1) being used for the padding value. 3D transpose convolution and batch normalization operations were used in compliance with three dimensions. A $32 \times 32 \times 32$ environment was used for the 3D depiction of voxel objects.

Fig. 1 2D sample object images from the ShapeNet dataset

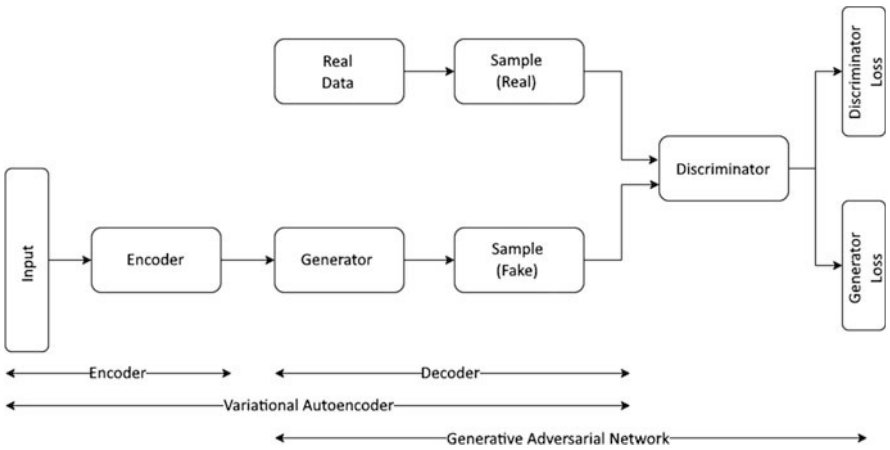

Fig. 2 3D-VAE-GAN model

The discriminator used comprises five layers. The first four layers include convolution, batch normalization, and LeakyReLU layers, while the final layer has a sigmoid layer immediately following the convolution process. For all layers, a kernel size of 4, a stride of 2, and (1,1,1,1) as the padding value were used. Convolution and batch normalization techniques were implemented in line with 3D. A $32 \times 32 \times 32$ environment was employed to create the 3D representation of voxel objects.

3 Pre-trained Network Models

Pre-trained networks can be defined as networks that have undergone pre-training on specific problems and already possess prior knowledge. Generally, these networks are pre-trained with substantial data corresponding to the problems they will address. They have the capability to incorporate the knowledge gained from previous data (known as transfer learning) while adapting to new tasks and problems. In essence, they involve two distinct training phases. Firstly, pre-trained networks undergo initial training on large datasets, followed by a second stage of training on specific, personalized problems. As pre-trained networks already possess knowledge of the problem, they achieve superior performance compared to networks trained from scratch. Additionally, pre-trained networks may possess a basic structure specific to the problem they are employed for. For instance, pre-trained networks employed for image classification may essentially involve a CNN model and units, whereas a pre-trained transformer is used in natural language processing.

DenseNet121, RegNet16, ResNet18, and EfficientNetB0 are among the pre-trained networks utilized in this study. Structural modifications were implemented in the pre-trained networks. Typically, the fully connected layers located at the end of the pre-trained networks were excised, and optimization was performed on the issue. The rationale for this customization is that such networks are generally tailored to particular problems, such as binary or multiclass classification, and output suitable values.

3.1 Densely Connected Convolutional Neural Networks 121 (DenseNet121)

DenseNet121 is a pre-trained convolutional neural network architecture extensively employed in computer vision tasks such as image classification and object recognition. It is a part of the DenseNet model family, which is comprised of "Densely Connected Convolutional Networks." DenseNet121, in particular, comprises 121 layers in its architecture. The dense connectivity pattern of DenseNet effectively resolves the vanishing gradient issue in deep neural networks. In this architecture, each layer is linked to every other layer in a feed-forward method. The dense connectivity enables enhanced feature reuse and information flow between layers, allowing for more effective training of deeper networks while using fewer parameters in comparison to conventional CNN structures such as VGG or ResNet. In Fig. 3, the structure of a standard DenseNet121 is shown.

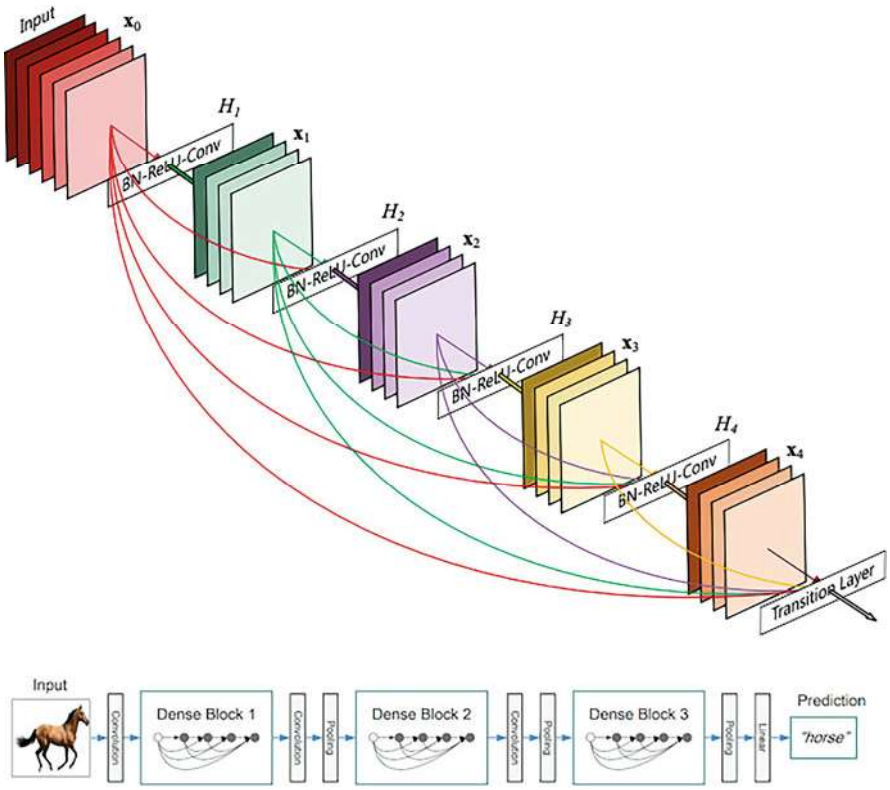

Fig. 3 Structure of a standard DenseNet121 pre-trained network [10]

3.2 Regular Networks 16 (RegNet16)

Regular network models optimize the structure of deep learning layers and computational resources. They are pre-trained to focus on two separate hyperparameters: the network's width, which is the number of filters (or neurons) in each layer, and the network's depth. Wider networks may yield better results, but with increasing width come higher computational costs and memory requirements. Network depth is determined by the number of layers in the network, and as depth increases, the learning capacity of the network may also increase, but at the expense of greater computational and memory demands. RegNet provides various approaches to optimizing network width and depth. When constructing RegNet models, various configurations of network width and depth are employed to obtain optimal values. As a result of these aspects, RegNet models can possess attributes such as minimal computational cost, low memory demand, and improved generalization.

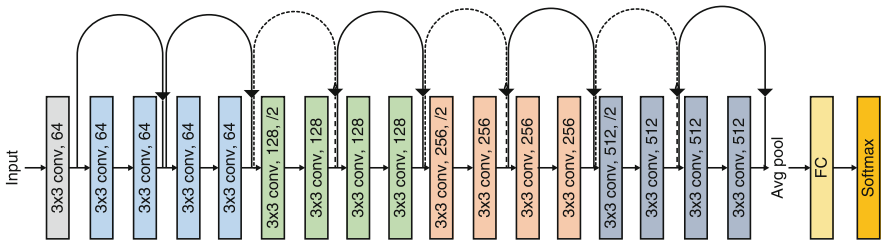

Fig. 4 Diagram of the ResNet18 model and its layers [13]

3.3 Residual Networks 18 (ResNet18)

It has been identified as a pre-trained network that facilitates deeper networks to achieve better performance by integrating the notion of residual learning. The number 18 in its name denotes the number of layers. These layers encompass convolution, normalization, and activation layers, along with residual connections. Residual connections enable input data to be directly added to the output when moving from one layer to another. This facilitates the backward propagation of gradients with greater ease and efficiency. Additionally, the network can be made deeper, thereby mitigating the risk of overfitting. The standard ResNet18 architecture is depicted in Fig. 4.

3.4 Efficient Networks B0 (EfficientNetB0)

This model, like others presented in this study, belongs to the deep learning family. The structure has designations such as B0, B1, and B2, with the number of layers typically expanding in line with the problem's complexity. Each of these adjustments is an optimized CNN model, aiming to deliver superior efficiency and performance. The primary objectives of such models are to attain heightened performance levels while reducing the number of parameters relative to larger and more intricate models. This is vital for systems with restricted computational resources and is specifically tailored for use on mobile devices. EfficientNet architectures are scalable, with Swish being the preferred activation function over ReLU in EfficientNetB0 due to its superior efficiency. While ReLU returns the input value of x if it is positive and 0 if it is negative, Swish returns a value in the format of x multiplied by sigmoid (x). Since weight sharing is employed in EfficientNetB0, it results in a reduction in the total amount of weight being used within the network. This approach scales the size and resources of every layer as per the size per channel, enabling every layer to function more effectively. Figure 5 typically depicts the EfficientNetB0 model.

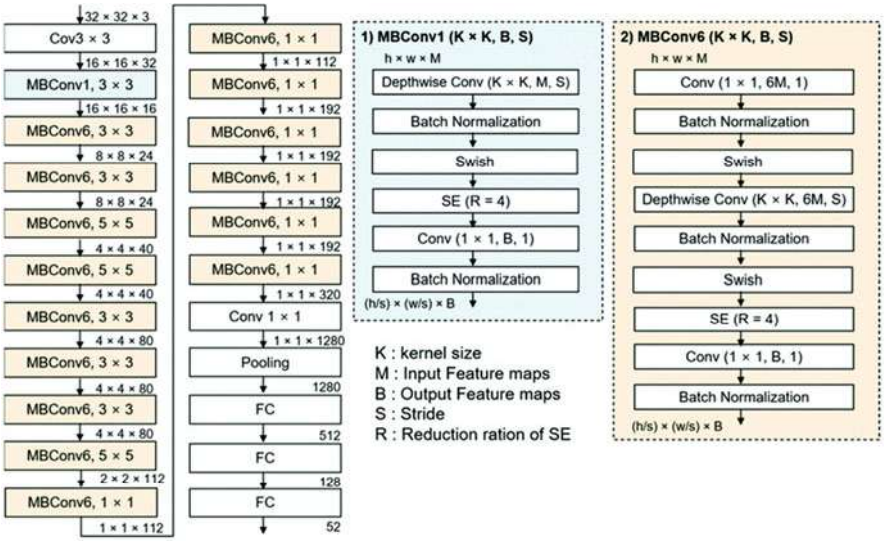

Fig. 5 The structure of the EfficientNetB0 model and the units it contains [26]

4 Results

In this study, the soft labeling method was employed to label both real and fake data. For labeling real data, a uniform distribution ranging from 0.7 to 1.2 was used, whereas a uniform distribution ranging from 0 to 0.3 was employed for fake data. The discriminator network learned at a faster pace compared to the generator network, and the training of the discriminator was halted when its accuracy surpassed 0.8 to prevent competitive behavior. The generator network and discriminator network utilized binary cross entropy as their loss function, while the encoder network utilized Kullback–Leibler divergence to gauge the likeness between the generated and real objects. The model's input consisted of 224 × 224 × 3-size images, which were converted to a 200-element latent space via the encoder network. Using the mapped data as input, the generator network created 32 × 32 × 32 3D voxel objects. The discriminator network determined whether these objects were real or fake, engaging in continuous competition in order to achieve optimization. The encoder network mapped 224 × 224 × 3 images to a 200-element latent space using pre-trained networks, which were then compared against each other. The study employed the Adam function and implemented a learning rate of 0.0025, 0.0001, and 0.001 for the generator, encoder, and discriminator networks, respectively, with a beta value of 0.5. A batch size of 32 was used. The training and evaluations were carried out on an i7-4790 CPU, GTX 980 Ti GPU, and 16 GB of RAM. The study utilized the PyTorch library and CUDA technologies [27].

Figure 6 shows the generator network loss, discriminator network accuracy and loss, Kullback–Leibler divergence, and total iteration time obtained as a result of

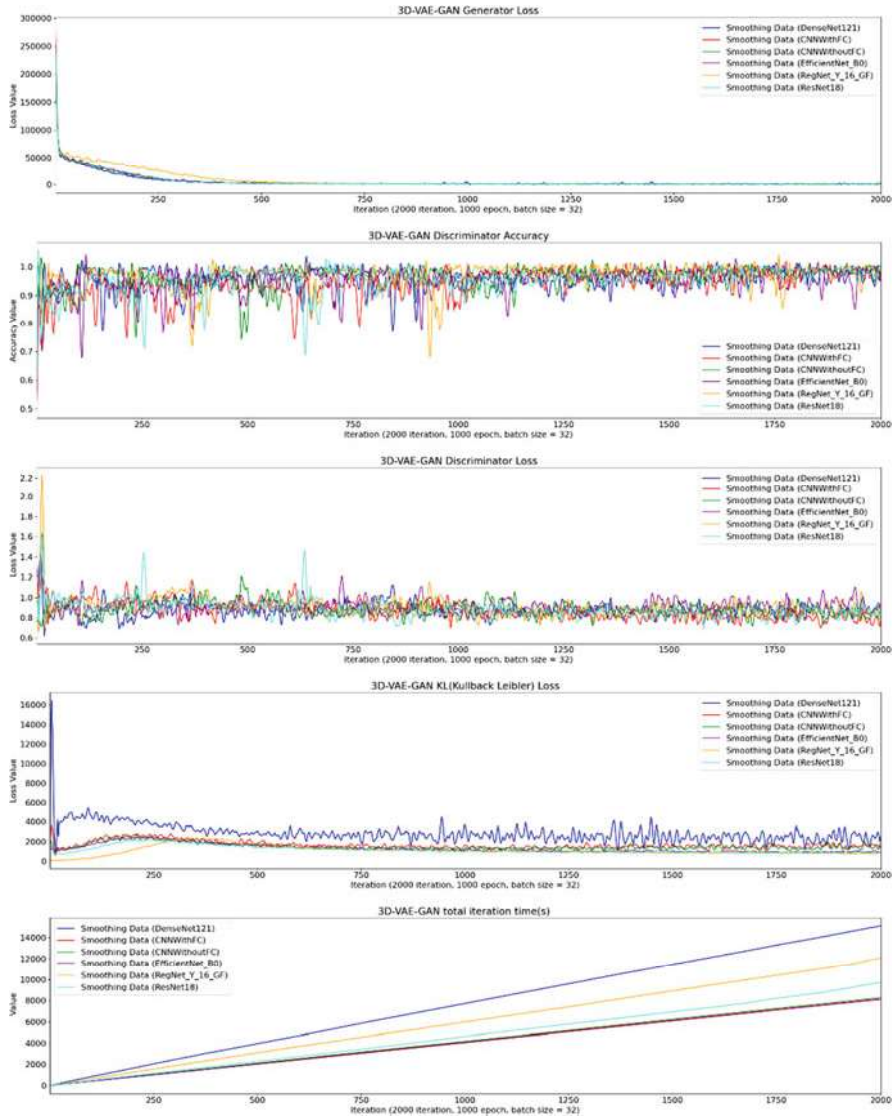

Fig. 6 Effect of pre-trained networks on performance metrics in encoder networks

training the encoder network with pre-trained networks. According to the obtained results, the loss of the generator network swiftly converges to zero. It is evident that there is a distinct disruption in the generator network loss for the RegNet16 model, unlike the other networks, up to 500 iterations. The analysis of discriminator network accuracy and loss values reveals similar outcomes. Upon examining the running times of the algorithms, it is noticeable that DenseNet121 operates the

slowest, followed by RegNet16 and then ResNet18. EfficientNetB0, the CNN with and without a fully connected layer, was faster than the other models and had similar running times among themselves. When assessing the Kullback–Leibler divergence value, which measures the similarity between the generated and real objects, it is evident that DenseNet121 and a CNN that lacks a fully connected layer obtain relatively lower results compared to other algorithms. ResNet18, RegNet16, and EfficientNetB0 achieve similar values. It is seen that the models start with a low Kullback–Leibler divergence value, which increases over time and then decreases steadily. This may be due to reasons such as the complexity of the model, learning speed, number of iterations, and data distribution.

Table 1 shows the changes in Kullback–Leibler divergence values of the models used throughout the epochs. To reach a more precise conclusion, Table 2 is given, which contains the mean, maximum, minimum, and standard deviations of the Kullback–Leibler values at the end of 1000 epochs. As can be seen in Table 2, RegNet16 gave the lowest value on the mean. Based on the mean values, RegNet16, ResNet18, and EfficientNetB0 can be ranked, respectively. However, if Fig. 6 is examined again, it will be seen that the order is exactly the opposite in terms of iteration time.

In this instance, if improved results for the problem are desired and time is not important to us, we should choose RegNet16; if time is very important to us and we want acceptable results, we should choose EfficientNetB0; and if we want an above-average value for both time and performance, we should choose ResNet18. It is advantageous to use these three pre-trained networks due to their superior performance as compared to other networks. As RegNet16 is considered superior in terms of performance, Fig. 7 displays the test results of the 3D-VAE-GAN model with the RegNet16 encoder. The upper part presents the 2D images provided as input, while the lower part showcases the 3D object outputs of 3D-VAE-GAN with RegNet16 as voxels.

5 Conclusions

Based on the findings of the study, an investigation into the application of 3D-VAE-GAN models in 3D object generation from 2D images was conducted. Specifically, the examination focused on utilizing several pre-trained convolutional neural networks (CNNs) as potential encoder networks in the VAE component. The examined pre-trained network models are DenseNet121, EfficientNetB0, RegNet16, and ResNet18, a standard CNN model with fully connected layers and a CNN model without fully connected layers. Feature extraction was executed within pre-trained network models, and the extracted features were then assigned to the latent space. The generator and discriminator networks of the GAN model were trained using the binary cross-entropy loss function, while the encoder network of the VAE model was trained using the Kullback–Leibler divergence to enhance the training process. Importantly, our methodology differs from traditional VAE models as we replace the

Table 1 Changes in the Kullback–Leibler divergence value during epochs

Epoch	Model name	Kullback–Leibler divergence value
1	DenseNet121	3215.43
	ResNet18	1662.01
	RegNet16	126.45
	EfficientNetB0	1367.98
	CNN with fully connected	924.78
	CNN without fully connected	2287.20
200	DenseNet121	2747.81
	ResNet18	1701.45
	RegNet16	2318.09
	EfficientNetB0	1936.44
	CNN with fully connected	2680.55
	CNN without fully connected	1959.02
400	DenseNet121	2403.31
	ResNet18	1253.69
	RegNet16	1106.81
	EfficientNetB0	1430.30
	CNN with fully connected	1536.77
	CNN without fully connected	1255.60
600	DenseNet121	2035.97
	ResNet18	1044.08
	RegNet16	986.19
	EfficientNetB0	1089.31
	CNN with fully connected	1561.37
	CNN without fully connected	1726.35
800	DenseNet121	3638.48
	ResNet18	950.78
	RegNet16	841.22
	EfficientNetB0	977.22
	CNN with fully connected	1766.02
	CNN without fully connected	1295.69
1000	DenseNet121	2534.85
	ResNet18	903.51
	RegNet16	909.86
	EfficientNetB0	924.78
	CNN with fully connected	1588.52
	CNN without fully connected	1772.49

decoder part of the VAE model with the generator network of the GAN model. We illustrate the duration of each iteration of the various algorithms and methods, along with the overall iteration time, using visual representations. Our focus during the training and testing phases is directed toward the category of “chair” contained within the ShapeNet dataset. As for the representation of 3D objects, we opted for a

Table 2 Mean, maximum, minimum, and standard deviation values of the Kullback–Leibler divergence value over 1000 epochs

Model name	Mean	Maximum	Minimum	Std. Dev.
DenseNet121	2893.807	30179.875	1079.060	1261.083
ResNet18	1219.067	2219.813	624.192	359.986
RegNet16	1129.660	2429.420	110.130	478.198
EfficientNetB0	1352.815	2579.661	876.164	424.084
CNN with fully connected	1696.749	4437.990	552.489	447.603
CNN without fully connected	1538.489	3579.641	679.022	396.773

Fig. 7 Test results of the 3D-VAE-GAN model with the RegNet16 encoder

voxel-based approach, which is well-suited for neural networks and deep learning processes. An indirect labeling technique is preferred over a direct one in determining an object's veracity. To ensure training stability, we impose constraints on the parser network to match the speed of the generator network. Our methodology uses a single 2D image as an input to create 3D objects. Based on our test outcomes, the pre-trained RegNet16 network exceeds other techniques, exhibiting exceptional efficiency in our setting.

The study indicates that utilizing a pre-trained network for the encoder network produces positive results. Future studies could investigate intervention not only in the encoder network but also in the GAN model's component. The generator and discriminator networks within the GAN structure may produce positive effects that, when combined with the findings of this study, enable the creation of successful models in both the VAE and GAN components. This may result in the production of overall more effective 3D-VAE-GAN models. Further research could explore diverse objects beyond chairs, as well as experimenting with point cloud and polygon approaches instead of voxel techniques. The use of multiple and cleaner images as input may also enhance performance.

References

1. Tai, M.C.-T.: The impact of artificial intelligence on human society and bioethics. Tzu-Chi Med. J. **32**, 339 (2020)
2. Moor, J.: The Dartmouth College artificial intelligence conference: the next fifty years. AI Mag. **27**, 87–87 (2006)
3. Russell, S.J.: Artificial intelligence a modern approach. Pearson Education, Inc., London (2010)
4. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT Press, Cambridge (2012)
5. Ongsulee, P.: Artificial intelligence, machine learning and deep learning. In: 2017 15th International Conference on ICT and Knowledge Engineering (ICT&KE), pp. 1–6. IEEE (2017)
6. Seo, Y., Dolan, R., Buchanan-Oliver, M.: Playing games: advancing research on online and mobile gaming consumption. Internet Res. **29**, 289–292 (2019)
7. Funkhouser, T.: Overview of 3d object representations. Princeton University, Princeton (2003)
8. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, p. 27, New York (2014)
9. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
10. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
11. Tan, M., Le, Q.: Efficientnet: rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105–6114. PMLR (2019)
12. Xu, J., Pan, Y., Pan, X., Hoi, S., Yi, Z., Xu, Z.: RegNet: self-regulated network for image classification. IEEE Trans. Neural Netw. Learn. Syst. **34**, 9562 (2022)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
14. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: a deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1912–1920 (2015)
15. Lim, J.J., Pirsiavash, H., Torralba, A.: Parsing ikea objects: fine pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2992–2999 (2013)
16. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: Advances in Neural Information Processing Systems, p. 29. MIT, Cambridge (2016)
17. Xie, J., Zheng, Z., Gao, R., Wang, W., Zhu, S.-C., Wu, Y.N.: Learning descriptor networks for 3d shape synthesis and analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8629–8638 (2018)
18. Wu, Z., Wang, X., Lin, D., Lischinski, D., Cohen-Or, D., Huang, H.: Sagnet: structure-aware generative network for 3d-shape modeling. ACM Trans. Graph. **38**, 1–14 (2019)
19. Wang, Y., Asafi, S., Van Kaick, O., Zhang, H., Cohen-Or, D., Chen, B.: Active co-analysis of a set of shapes. ACM Trans. Graph. **31**, 1–10 (2012)
20. Sidi, O., Van Kaick, O., Kleiman, Y., Zhang, H., Cohen-Or, D.: Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. In: Proceedings of the 2011 SIGGRAPH Asia Conference, pp. 1–10 (2011)
21. Van Kaick, O., Tagliasacchi, A., Sidi, O., Zhang, H., Cohen-Or, D., Wolf, L., Hamarneh, G.: Prior knowledge for part correspondence. In: Computer Graphics Forum, pp. 553–562. Wiley Online Library, Hoboken (2011)
22. Guan, Y., Jahan, T., van Kaick, O.: Generalized autoencoder for volumetric shape generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 268–269 (2020)

23. Wu, R., Zhuang, Y., Xu, K., Zhang, H., Chen, B.: Pq-net: a generative part seq2seq network for 3d shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 829–838 (2020)
24. Deng, J., Shi, S., Li, P., Zhou, W., Zhang, Y., Li, H.: Voxel r-cnn: towards high performance voxel-based 3d object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1201–1209 (2021)
25. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H.: Shapenet: an information-rich 3d model repository. arXiv preprint arXiv:1512.03012. (2015)
26. Gang, S., Fabrice, N., Chung, D., Lee, J.: Character recognition of components mounted on printed circuit board using deep learning. Sensors. **21**, 2921 (2021)
27. Nickolls, J., Buck, I., Garland, M., Skadron, K.: Scalable parallel programming with cuda: is cuda the parallel programming model that application developers have been waiting for? Queue. **6**, 40–53 (2008)